Master's Thesis

Research on the Computational Complexity of Walls and Warriors Puzzle

2410113 ZHAO, Shuai

Supervisor UEHARA, Ryuhei

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

February 2026

Abstract

The Walls & Warriors (W&W) puzzle is a two-dimensional grid-based logic puzzle for one player. The player must place wall segments to form a single closed loop that separates "Blue Warriors" (inside) from "Red Warriors" (outside). This thesis investigates the computational complexity of W&W through a rigorous graph-theoretic formulation, transforming the puzzle into a decision problem of finding a simple cycle under spatial separation constraints. Specifically, we model the game board as a graph where wall placements act as edge selections, requiring the loop to topologically isolate specific vertex sets based on their chromatic designation.

Our main contributions are twofold. First, we prove that the general W&W decision problem is **NP-complete**. Membership in NP is established by showing that a candidate solution can be verified in polynomial time. NP-hardness is demonstrated via polynomial-time reductions from two known NP-complete problems: Planar Tree-Residue Vertex-Breaking (TRVB) and Rectilinear Steiner Tree (RST). These reductions employ carefully designed gadgets that encode topological and geometric constraints into the puzzle's loop-formation mechanics; the TRVB reduction captures the difficulty of navigating logical obstacles by mapping the connectivity states of graph vertices to local wall configurations that can either connect or disconnect adjacent regions. Conversely, the RST reduction addresses the complexity of geometric resource optimization, demonstrating that finding a valid loop enclosing specific points mirrors the minimum-length constraints inherent in Steiner trees.

Second, we identify a tractable case: when the grid height is bounded, the problem becomes **Fixed-Parameter Tractable (FPT)** with respect to the height. We develop a dynamic programming algorithm that processes the grid vertex-by-vertex. By maintaining a complex frontier state and scanning the grid in a top-to-bottom, left-to-right order, the problem can be solved efficiently on narrow grids. The state representation at each step involves a tuple (P, U, C, Q), where P tracks the connected components of wall segments crossing the boundary (plugs), U indicates vertical connections, C is a bitmask dynamically tracking the inside/outside chromatic status of regions, and Q serves as a global flag to enforce the single-loop constraint. This precise state management allows the algorithm to achieve a time complexity of $O(W \cdot H \cdot 8^H)$. This confirms that W&W is solvable in linear time relative to width W for a fixed height H.

The research demonstrates that despite its deceptively simple ruleset, the W&W puzzle exhibits substantial computational complexity. Our findings contribute to complexity theory for grid puzzles and offer insights into enforcing topological constraints.

Keywords: Computational Complexity, NP-completeness, Dynamic Programming, Fixed-Parameter Tractability, TRVB, RST, Grid Puzzles, Cycle Formation, Geometric Constraints.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Background

Puzzles and games have long been a fertile ground for exploring computational complexity, as they often encapsulate real-world problems in abstract and intuitive forms [1]. The study of these puzzles provides insights into algorithm design and the boundaries of computability. While some puzzles can be solved efficiently in polynomial time (class **P**) when their configurations are sufficiently restricted, many others have been proven to be intractable.

For instance, path-finding puzzles on grid graphs often relate to the *Hamiltonian Path* problem, which is known to be **NP-complete** [2]. Similarly, edge-matching and connection puzzles, such as *Pipe Puzzles*, have been rigorously analyzed by Shirayama et al., who demonstrated their **NP-completeness** and explored polynomial-time solutions under restricted conditions [3].

Moreover, puzzles involving moving components, such as general *Sliding Block Puzzles*, are known to be **PSPACE-complete** [4]. In these cases, even on a board of fixed size, the number of moves required to reach a solution can be exponential relative to the board dimensions, making the state space practically vast despite being theoretically finite.

Beyond these complexity classes lies the realm of *undecidability*. This typically arises in puzzles defined on infinite grids or involving an infinite supply of components, such as the *Tiling Problem* [5, 6]. In such scenarios, the search space is truly unbounded, rendering it mathematically impossible to determine, in the general case, whether a solution exists.

1.2 Walls & Warriors Puzzle

Figure 1.1: Walls & Warriors puzzle (Source: `https://www.smartgamesusa.com/sites/default/files/smartgames-product-banner_Walls%26Warriors_0.jpg`)

The *Walls & Warriors* (W&W) puzzle, as shown in Figure 1.1, is a 2D grid-based logic board puzzle for one player [7]. The objective is to arrange a set of wall segments to make a fully-enclosed castle. Specifically, in the physical version, the player must place all four wall pieces to form a single enclosed region that satisfies two geometric constraints:

1. **Inclusion:** All "blue warriors" (friendly units) and the high tower must be contained within the *interior* of the wall configuration.
2. **Exclusion:** All "red warriors" (enemy units) must be strictly excluded *outside* the wall configuration.

As shown in Figure 1.2, the physical components include a game board, 4 wall pieces of varying shapes, two types of units: blue warriors and red warriors, and a blue tower. The wall pieces are designed to fit onto the grid

edges, while the warriors occupy the cells. In our mathematical model, these will be abstracted as sets of edges and faces, respectively.

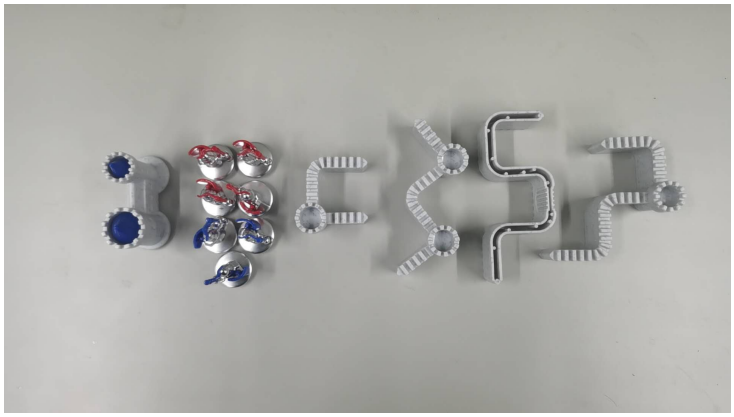


Figure 1.2: Physical components of the W&W puzzle

While grid-based pathfinding is well-studied, W&W's unique combination of *loop formation* and *chromatic separation* introduces novel topological constraints. To date, the computational complexity of this specific constraint combination in the W&W puzzle remains unclassified in established complexity classes.

1.3 Motivation and Comparison

To highlight the uniqueness of the W&W puzzle, we differentiate it from two classic NP-complete grid puzzles: Slitherlink and Yin-Yang.

- **Slitherlink** focuses on forming a single closed loop based on local numeric clues. While it shares the loop formation aspect, it lacks any requirement for point separation or containment based on specific unit types [8].
- **Yin-Yang** requires partitioning the grid into two connected monochromatic regions. However, its connectivity constraints are purely regional rather than topological regarding containment [9].

Unlike these similar grid puzzles, W&W is defined by the spatial partition of the entire board where the loop must correctly contain or exclude various types of units. This combination of loop formation and chromatic separation introduces a unique layer of complexity not fully captured by existing grid puzzle models.

1.4 Objectives

This research aims to rigorously formalize the W&W puzzle as a mathematical decision problem and classify its computational complexity. The study explores the boundary between its intractability and tractability through the following objectives:

- **Analyzing Intractability:** We aim to establish the computational complexity of the general W&W problem by demonstrating that it belongs to a known intractability class (NP-complete). This will be achieved by constructing a polynomial-time reduction from two known NP-complete problems.
- **Exploring Tractability:** We seek to identify specific restrictions under which W&W becomes efficiently solvable. In particular, we investigate algorithmic techniques such as *dynamic programming* to design polynomial-time algorithms for instances with bounded structural parameters, proving its *fixed-parameter tractability* (FPT).

1.5 Structure of the Thesis

The remainder of this thesis is organized as follows:

- **Chapter 2**: We provide the necessary formal definitions of grid graphs and introduce two known NP-complete problems (TRVB, RST) used for our subsequent reductions. Furthermore, we define Fixed-Parameter Tractability (FPT) to lay the groundwork for our tractable parameter analysis.
- **Chapter 3**: We establish the formal mathematical model of the W&W puzzle. We translate the physical game components into graph-theoretic terms and rigorously define the constraints governing a valid solution.
- **Chapter 4**: We present the core complexity analysis and prove that the W&W decision problem is NP-complete through polynomial-time reductions. This chapter demonstrates that the puzzle is intractable even on simple grid structures.
- **Chapter 5**: We explore tractable cases where the problem becomes tractable. We propose a Fixed-Parameter Tractable (FPT) algorithm based on dynamic programming, achieving $O(W \cdot H \cdot 8^H)$ time complexity for bounded grid height.
- **Chapter 6**: We summarize our findings in puzzle complexity and parameterized algorithms.

Chapter 2

Preliminaries

In this chapter, we establish the fundamental concepts and notations used throughout this thesis. We begin by defining the underlying graph-theoretic structures, followed by two known NP-complete problems, which serve as the theoretical basis for our hardness proofs. Then, we introduce the framework of parameterized complexity, which is essential for analyzing the tractability of the W&W puzzle under specific structural constraints.

2.1 Grid Graph

The underlying structure of the W&W puzzle is defined on a grid. To model the finite game board rigorously, we consider it as a subgraph of the infinite integer grid $\mathbb{Z}^2$.

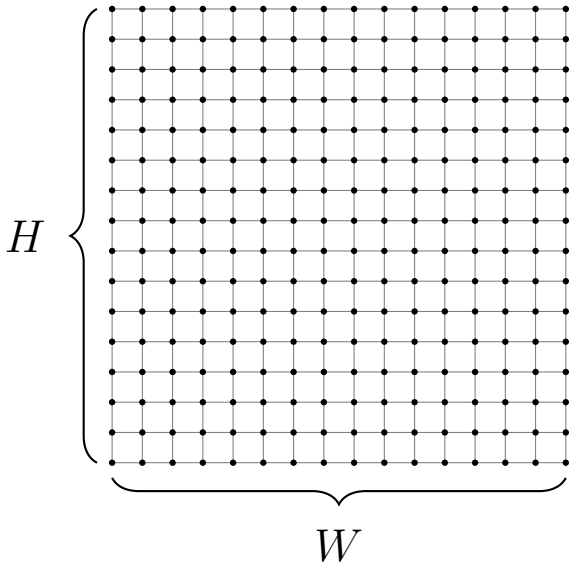


Figure 2.1: Grid Graph

As illustrated in Figure 2.1, a grid graph $G = (V, E)$ is defined as a finite

induced subgraph of $\mathbb{Z}^2$. Formally, for a board of height H and width W, the vertex set is defined as:

$$V = \{(i, j) \in \mathbb{Z}^2 \mid 0 \leq i \leq H, 0 \leq j \leq W\} \tag{2.1}$$

where i denotes the row index and j denotes the column index.

The edge set E consists of connections between adjacent vertices. Two vertices $u, v \in V$ are connected by an edge if and only if their L_1 distance is exactly 1:

$$E = \{\{u, v\} \subseteq V \mid ||u - v||_1 = 1\} \tag{2.2}$$

Additionally, we define the set of faces F corresponding to the grid cells. We identify each face by the coordinate of its top-left vertex, formally defined as the discrete set:

$$F = \{(i, j) \in \mathbb{Z}^2 \mid 0 \leq i < H,\ 0 \leq j < W\} \tag{2.3}$$

Then, for a face $u = (i, j) \in F$, cell(u) represents the open unit square region:

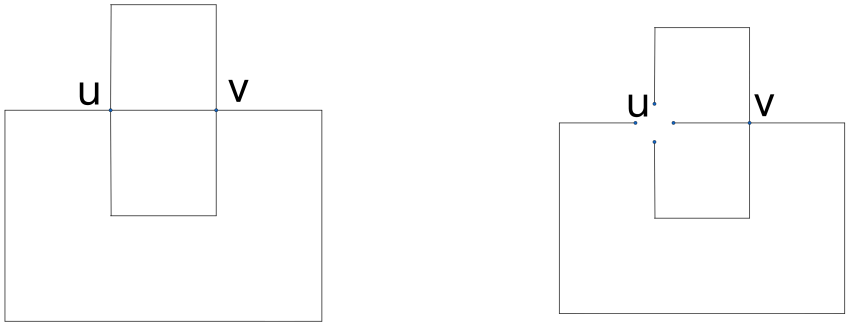
$$\text{cell}(i, j) = \{(x, y) \in \mathbb{R}^2 \mid j < x < j + 1,\ i < y < i + 1\} \tag{2.4}$$

2.2 Related Computational Problems

The hardness proofs presented in Chapter 4 rely on reductions from two known NP-complete problems. In this section, we formally define the computational problems that provide the theoretical basis for this transformation.

2.2.1 Tree-Residue Vertex-Breaking (TRVB)

Tree-Residue Vertex-Breaking (TRVB), introduced by Demaine and Rudoy [10], is a primary tool for proving the hardness of grid-based puzzles involving loop connectivity constraints. Specifically, the TRVB problem serves as a robust alternative to traditional reductions from the Hamiltonian Cycle problem. Additionally, Demaine and Rudoy proved that TRVB remains NP-complete even when restricted to degree-4 breakable vertices [10].



(a) TRVB instance (b) The solution of TRVB instance

Figure 2.2: Illustration of a minimal TRVB instance

In this thesis, we utilize the standard planar 4-regular variant, where the input graph has degree 4 and vertices are all marked as breakable. Here is the formal definition of the problem:

Problem 1: ***Planar 4-regular Tree-Residue Vertex-Breaking***

Input: A planar 4-regular graph $G = (V, E)$.
Question: Is there a subset of vertices $V' \subseteq V$ such that breaking each vertex in V' results in a graph that is a single tree?

To illustrate, Figure 2.2a shows a minimal multigraph instance with two vertices, u (left) and v (right), connected by four parallel edges. Suppose we designate the left vertex to be broken and the right vertex v keeps the unbroken state. A solution requires finding a breaking configuration for u that transforms the multigraph into a connected tree structure without cycles as shown in Figure 2.2b.

2.2.2 Rectilinear Steiner Tree (RST)

To address the resource constraints (specifically the limited number of wall pieces), we utilize the Rectilinear Steiner Tree problem [11]. To illustrate, Figure 2.3 and Figure 2.4 shows an instance of RST and its solution, where the objective is to connect a given set of points (terminals) in a grid using the shortest possible total edge length, allowing the addition of extra points (Steiner points) to minimize the overall length.

Here is the formal definition of the problem:

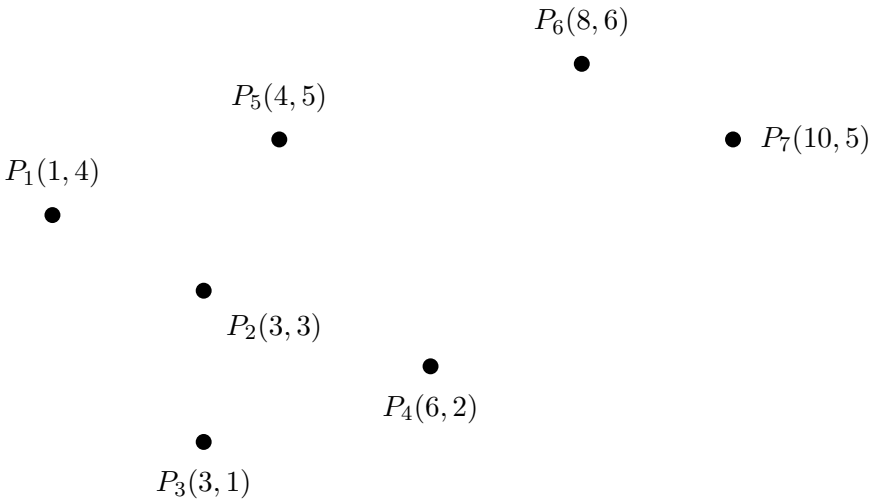


Figure 2.3: An instance of RST and a length constraint $\ell = 19$.

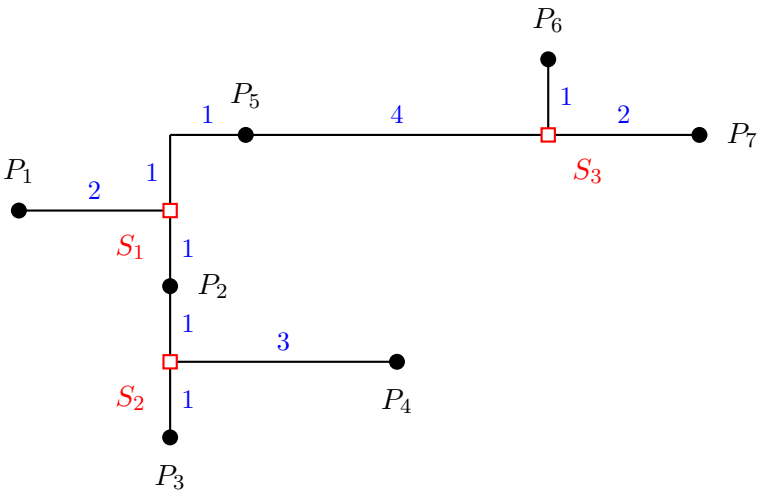


Figure 2.4: Solution to the RST instance with total length constraint $\ell = 17$. The square nodes represent the Steiner points used to minimize the total edge length.

Problem 2: *Rectilinear Steiner Tree*

Input: A set of points $P \subset \mathbb{Z}^2$ in a grid graph $G = (V, E)$ of size $H \times W$, and an integer L
Question: Does there exist a tree in G of total length at most L that connects all points in P?

2.3 Parameterized Complexity

To analyze the algorithmic tractability of the W&W puzzle under structural constraints, we employ the framework of parameterized complexity. Unlike classical complexity theory, which measures runtime solely in terms of the input size n, parameterized complexity measures runtime in terms of both n and a parameter k.

2.3.1 Fixed-Parameter Tractable

A problem with input size n and a parameter k is said to be **Fixed-Parameter Tractable (FPT)** if it can be solved in time:

$$O(f(k) \cdot n^c) \tag{2.5}$$

where:

- n is the size of the input.
- c is a constant independent of k.
- $f(k)$ is an arbitrary computable function that depends only on the parameter k.

This definition implies that for a fixed parameter k, the problem is solvable in polynomial time with respect to the input size $|x|$. In the context of this thesis, we consider the grid width W as the input size and the grid height H as the parameter k.

Chapter 3

Mathematical Modeling

In this chapter, we provide a rigorous formalization of the W&W puzzle, translating its physical game mechanics into precise geometric and graph-theoretic constraints.

3.1 Generalized W&W puzzle

The physical version of the W&W puzzle typically consists of a fixed-size board, as shown in Figure 3.1, and a limited set of puzzle pieces. To analyze the problem from the perspective of computational complexity, we must generalize these parameters. We define a generalized W&W puzzle as one played on an $W \times H$ rectangular grid, where $W, H \in \mathbb{N}$ can be arbitrarily large. The number and placement of blue and red warriors are also unrestricted, allowing for any configuration on the grid. This generalization enables us to explore the problem's complexity in a broader context, beyond the constraints of a specific physical instance.

Figure 3.1: Game board of the W&W puzzle

3.2 Game Board

The game board is modeled as a finite rectangular grid graph $G = (V, E)$, utilizing the formal definitions established in Section 2.1. Each cell in the grid corresponds to a face in the graph, denoted as F. The vertices V represent the intersection points of the grid lines, while the edges E represent the line segments connecting these vertices.

3.3 Puzzle Elements and Spatial Relations

To rigorously bridge the physical game mechanics with our graph-theoretic formulation, we abstract the physical wall pieces of varying shapes into standardized **unit-length edges**. This abstraction allows us to define the puzzle on generalized grid graphs and precisely quantify the resources available to the player.

Input: A tuple $\mathcal{I} = (G, B, R, S)$, where:

- **Game Board** G: A grid graph representing the game board.
- **Blue Warriors ($B \subseteq F$):** A set of faces representing friendly units that must be enclosed.
- **Red Warriors ($R \subseteq F$):** A set of faces representing enemy units that must be excluded.
- **Wall Resources** S: A set of unit-length wall segments placed on G.

Output (Wall Configuration C): A subset of walls $C \subseteq S$ forms a simple cycle on G. Any simple cycle C partitions the plane $\mathbb{R}^2$ into two disjoint regions: a bounded **interior** $\mathrm{Int}(C)$ and an unbounded **exterior** $\mathrm{Ext}(C)$.

Note that we omit the physical **High Tower** as a unique component and instead treat it as two blue warriors placed in adjacent cells. This simplification is justified because any solution that encloses two adjacent blue cells effectively satisfies the constraints of the tower. Since our complexity analysis in Chapter 4 focuses on the general case with arbitrary warrior placements, this abstraction does not lose generality and simplifies the formal proofs.

3.4 Constraints

Recall the geometric mapping of cells defined in Section 2.1. For any $b \in B$ and $r \in R$, the notations $\mathrm{cell}(b)$ and $\mathrm{cell}(r)$ represent the specific geometric

regions where the blue and red warriors are placed, respectively. We define the constraints that a valid wall configuration C must satisfy:

1. **Loop Constraint:** The edge set C placed on G must induce a single non-self-intersecting simple cycle.
2. **Separation Constraint:** All cells containing blue warriors must reside within the bounded region:

$$\forall b \in B, \quad \text{cell}(b) \subseteq \text{Int}(C) \tag{3.1}$$

   Simultaneously, all cells containing red warriors must reside in the unbounded region:

$$\forall r \in R, \quad \text{cell}(r) \subseteq \text{Ext}(C) \tag{3.2}$$

3. **Connectivity Constraint:** To prevent "internal voids" that bypass the game's logic, all regions belonging to $\text{Ext}(C)$ must be path-connected to the infinite outer boundary of the board. This ensures that the "enemy" territory is a single contiguous space.

3.5 Decision Problem

The primary objective of this thesis is to classify the computational complexity of finding such a wall configuration. To accommodate different computational complexity analyses in Chapter 4, we distinguish between two variants of the W&W puzzle:

Problem 3: *1-Loop W&W*

Input: A tuple (G, B, R, S).
Question: Does there exist a wall configuration $C \subseteq S$ that satisfies all constraints?

Problem 4: *Bounded 1-Loop W&W*

Input: A tuple (G, B, R, S), with $|S| \leq \ell'$ for some integer ℓ'.
Question: Does there exist a wall configuration $C \subseteq S$ that satisfies all constraints with perimeter bound ℓ'?

Chapter 4

Intractability

In this chapter, we establish the computational intractability of the generalized W&W puzzle. We begin with a reduction from **Planar Tree-Residue Vertex-Breaking (TRVB)** to establish the fundamental *topological complexity* of loop formation and region separation. Subsequently, we employ **Rectilinear Steiner Tree (RST)** to further bound the problem within the contexts of *geometric resource optimization*.

4.1 Reduction from TRVB to W&W

To establish the fundamental topological complexity, we first reduce from Problem 1.

Theorem 1

The generalized 1-loop Walls & Warriors puzzle is NP-complete.

Conceptually, the W&W puzzle can be modeled as the geometric expansion of an abstract graph embedded on the grid: vertices and edges are expanded into interconnected grid regions, and the final solution loop constitutes the boundary of a specific polyomino. From this perspective, the TRVB problem maps naturally onto the W&W puzzle, where "breaking" a vertex corresponds to altering the connectivity of the enclosed region.

For this specific reduction, we analyze the puzzle under a relaxed resource constraint: we assume the player has access to an unlimited supply of wall segments of arbitrary shapes.

In the following sections, we detail the construction of the reduction, the design of the necessary gadgets, and the proof of equivalence between the two problems.

4.1.1 Gadgets

The design of our gadgets draws much inspiration from the reduction techniques used by Demaine et al. for the **Yin-Yang** puzzle [9]. However, we adapt and extend these concepts to fit the unique mechanics of the Walls & Warriors puzzle, particularly focusing on the wall placement constraints and the separation of warrior types. Note that all gadgets are allowed to be rotated and mirrored as necessary to fit the global grid layout.

4.1.1.1 Vertex Gadgets

As illustrated in Figure 4.1, we employ a special 7×7 tiling design for the vertex gadget to represent the vertices of G. This gadget acts as a 4-way junction connecting incident edge gadgets.

To maintain global alignment, the gadget accepts horizontal connections strictly at row 3 and vertical connections at column 4. This ensures that when a vertex gadget is placed adjacent to any edge gadget, the "corridors" of connectivity meet seamlessly at the boundaries.

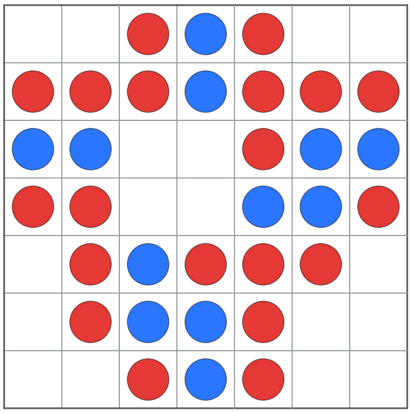
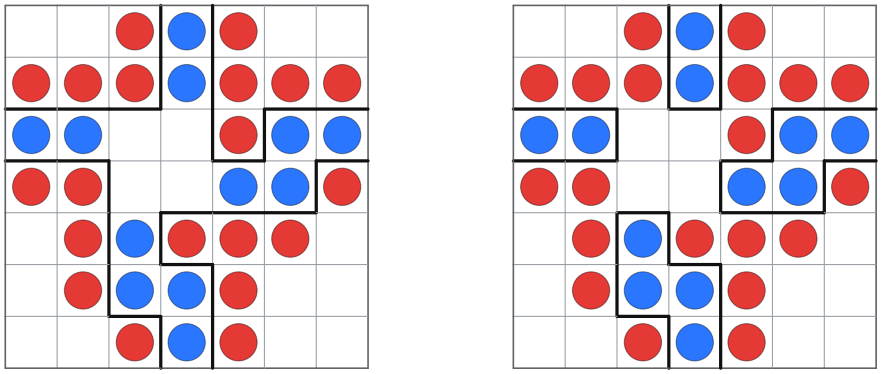


Figure 4.1: Vertex Gadget for TRVB reduction

The internal arrangement of warriors is constructed to support exactly two topological states, corresponding to the "breakable" nature of the TRVB problem. The dense placement of red and blue Warriors within the 7×7 block leaves no room for deviation, enforcing a unique valid configuration for each state, as shown in Figure 4.2:



(a) Unbroken State (b) Broken State

Figure 4.2: The two states of the Vertex Gadget, showing the unique valid configuration for each.

- **State 1: Unbroken (Connected)**
  In this state, the wall encloses the center of the gadget. This configuration connects all four incident edge corridors into a single component within the gadget. There is exactly one valid way to place the wall segments to achieve this enclosed state without violating the separation constraints.
- **State 2: Broken (Disconnected)**
  In this state, the wall bypasses the center, effectively splitting the junction. This configuration severs the connection between the incident edges, simulating the disconnection of the vertex from the connected component. Similarly, there is exactly one valid wall configuration that satisfies the "broken" topology.

To rigorously establish the uniqueness of these configurations, we present the following lemma.

Lemma 1

As shown in Figure 4.2, for any vertex gadget, there exist exactly two valid local wall configurations, corresponding to the unbroken and broken states.

Proof. The proof proceeds by decomposing the gadget into two structural components: the peripheral **rigid channels** and the central **decision region**.

1. Rigidity of Peripheral Channels. Each of the four channels connecting to the gadget is constructed using a dense alternating arrangement of blue and red warriors. By the separation constraint, the wall configuration C must strictly follow the unique middle line between adjacent blue and red warriors. Any deviation would immediately intersect a warrior-occupied cell, violating the separation constraint. Consequently, within each channel, the wall's trajectory is uniquely determined.
2. Central Decision Region (2×2 empty area). The four rigid channels converge at a central 2×2 region of empty cells, denoted Ω. Formally, considering the planar embedding, the four entry points on the boundary of Ω impose strict parity constraints that any routing other than the "Unbroken" or "Broken" configurations would require the path to either self-intersect within Ω or leave a channel unconnected, violating the single-loop constraint.
Conclusion. Since the peripheral channels enforce a unique path leading into Ω, and Ω itself admits only two valid configurations that satisfy all constraints, we conclude that the vertex gadget has exactly two valid local wall configurations: the unbroken and broken states. □

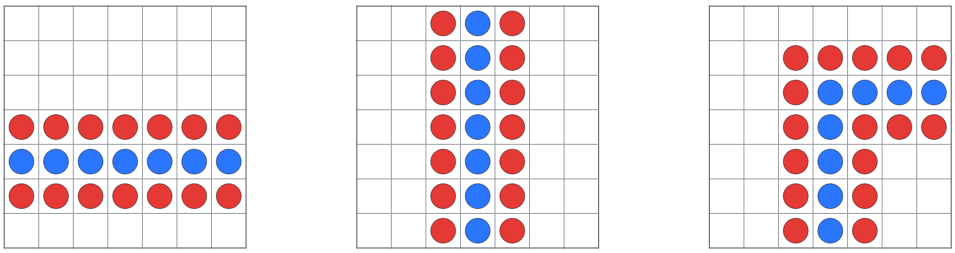
4.1.1.2 Edge Gadgets

Edges in the planar graph G are modeled as "corridors" that transmit connectivity between vertices. To ensure consistent alignment across the entire grid, every gadget in our construction is also standardized to a fixed size of 7×7 cells. To accommodate the rectilinear embedding of G on the grid, we utilize three specific variants of edge gadgets, as illustrated in Figure 4.3:

- **Horizontal Gadgets:** Transmit connectivity horizontally. The blue warriors form a continuous line along row 3.
- **Vertical Gadgets:** Transmit connectivity vertically. The blue warriors form a continuous line along column 4.
- **Corner Gadgets:** Facilitate 90-degree turns, connecting a horizontal input at row 3 to a vertical output at column 4.

4.1.2 Hardness Proof

To visualize the global consistency of our reduction, Figure 4.4 and Figure 4.5 illustrate a complete W&W instance constructed from a 2-vertices TRVB instance shown in Figure 2.2 and a corresponding valid solution configuration. Furthermore, the entire empty space of the grid surrounding the vertex and edge gadgets is filled with red warriors. This ensures that the wall



(a) (b) (c)

Figure 4.3: The three types of edge gadgets used to make connections across the grid. Showing (a) Horizontal, (b) Vertical, and (c) Corner variants.

configuration remains confined within the intended gadget area, establishing a consistent exterior region.

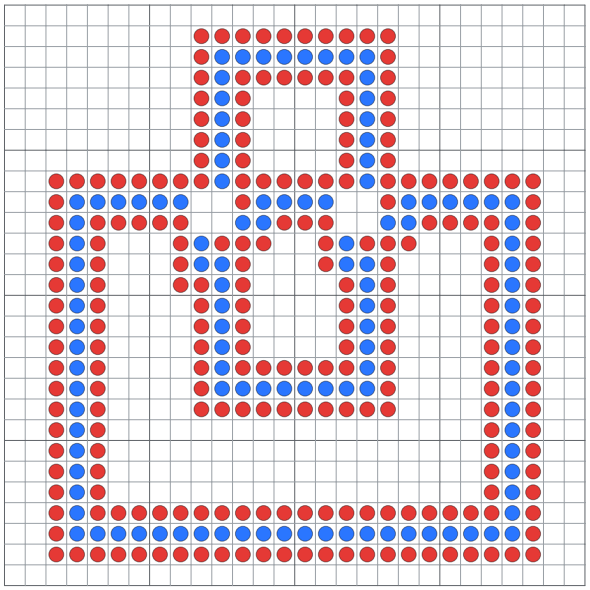


Figure 4.4: W&W from TRVB Instance

By using the gadget constructions described above, we can analyze the complexity of the reduction and verify that the W&W is indeed NP-complete.

Proof. To prove NP-completeness, we must show that the problem belongs to the class NP and that it is NP-hard.

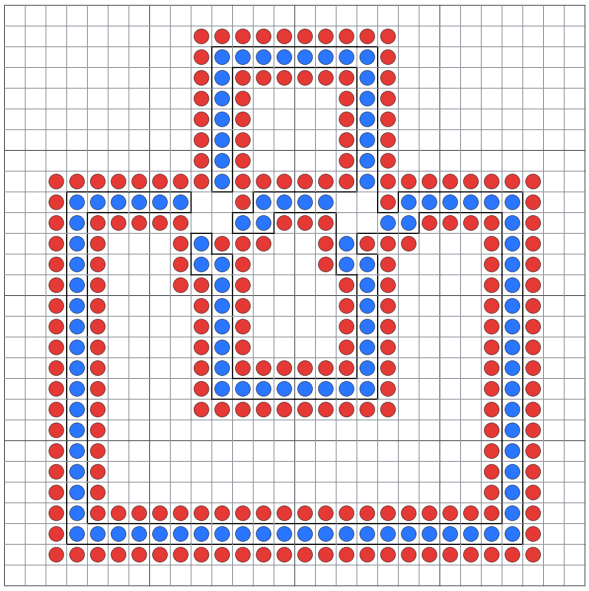


Figure 4.5: The solution of W&W from TRVB Instance

1. Membership in NP First, we verify that a candidate solution can be checked efficiently in polynomial time with respect to the input size. Given a W&W instance and a candidate wall configuration C (a set of edge segments):

- **Loop Constraint:** Verifying that C forms a single, non-intersecting closed loop can be performed using a standard graph traversal (e.g., DFS) in $O(|V| + |E|) = O(n)$ time.
- **Separation Constraint:** Checking that all blue warriors are in $\text{Int}(C)$ and all red warriors are in $\text{Ext}(C)$ can be done via a flood-fill algorithm in $O(|V| + |E|) = O(n)$ time.
- **Connectivity Constraint:** Verifying that the red region is connected to the grid boundary requires a BFS starting from the boundary cells, computable in $O(n)$ time.

Since we can verify all constraints in polynomial time, the generalized 1-loop W&W puzzle is in NP.

2. NP-Hardness We establish NP-hardness by constructing a reduction from Problem 1 by using these gadgets. The reduction proceeds by first embedding the input planar graph onto a rectilinear grid. Utilizing the algorithm proposed by Papakostas and Tollis [12], we generate an orthogonal drawing of the graph in linear time, which occupies an area quadratic in the number of vertices ($O(n^2)$). This ensures that the entire reduction

process—from layout generation to gadget placement—is performed in polynomial time relative to the number of vertices in the TRVB instance.

The correctness of the reduction is established by the topological correspondence between the graph structure and the grid regions.

- **Forward Direction:** A valid tree residue in TRVB is a connected, acyclic subgraph. When mapping to the grid, this ensures the "Inside" region (blue warriors) is connected and contains no "holes" (internal red voids). The boundary of such a hole-free connected region is guaranteed to be a single simple loop, satisfying the W&W constraints.
- **Reverse Direction:** Suppose the constructed W&W instance admits a valid wall configuration C. By the connectivity constraint of the puzzle, all cells containing red warriors must belong to a single connected exterior region that leads to the boundary of the grid. Consequently, the region enclosed by C is a finite, simply connected region with no internal holes. By Lemma 1, the local wall configuration within each vertex gadget is strictly limited to either an "Unbroken" or "Broken" state. This creates a direct mapping, where each "Unbroken" gadget corresponds to a remaining vertex in the graph. Consider the structure induced by these remaining vertices. Since the interior region is connected, the corresponding vertices form a connected subgraph. Crucially, if these vertices were to form a cycle, the corresponding closed loop of gadgets would enclose a bounded region in the grid that is disconnected from the exterior. Such a region would necessarily contain red cells that are isolated from the boundary, violating the separation constraint of the puzzle. Therefore, no cycle can exist in the induced subgraph. It follows that the connected structure of active vertices is acyclic. A connected, acyclic subgraph is, by definition, a tree residue. This establishes a valid solution to the TRVB instance.

Thus, the W&W instance admits a valid solution if and only if the TRVB instance has a valid vertex-breaking set. Then the generalized 1-loop W&W puzzle is NP-hard.

Conclusion Since the problem is in NP and is NP-hard via reduction from TRVB, the generalized 1-loop W&W puzzle is NP-complete. □

4.2 Reduction from RST to W&W

Following the topological analysis using TRVB, we now address the geometric optimization aspect of the W&W puzzle. We reduce from the Problem 2

to demonstrate that finding a solution even under strict wall perimeter constraints is still NP-complete.

Theorem 2

The generalized 1-loop Walls & Warriors puzzle is NP-complete even with a perimeter length limit ℓ'.

Conceptually, this reduction maps the "total length" minimization in RST to the "perimeter" minimization in W&W. We construct a W&W instance where the grid is scaled by a factor of 3. In this expanded grid, the wall configuration must traverse the corridors back and forth to enclose the required terminals, effectively doubling the path length relative to the underlying Steiner tree. Specifically, a Steiner tree of length ℓ corresponds to a wall configuration of perimeter $6\ell + 4$.

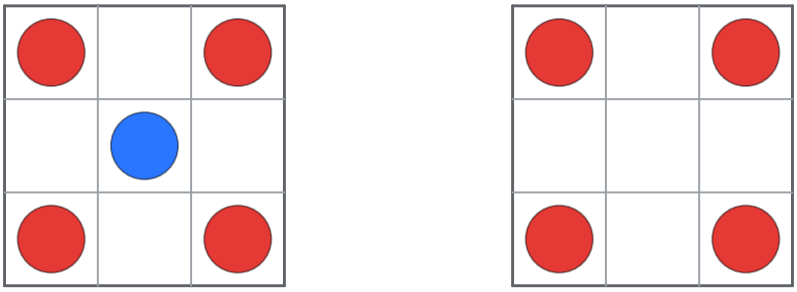
4.2.1 Gadgets

To implement this reduction, we design specific gadgets that strictly enforce the rectilinear distance metric (L_1 norm). All gadgets are standardized to a 3×3 size to ensuring seamless tiling, similar to the construction in Section 4.1.

4.2.1.1 Vertex Gadgets

To represent the terminals points of the original instance, we define **Terminal Vertex Gadgets** and **Empty Vertex Gadgets**, respectively. As shown in Figure 4.6, the former contains a central blue warrior, while the latter is devoid of any inclusion units. To ensure geometric fidelity, red warriors are employed to restrict wall segments strictly to the corridors within 4-way junctions.

- **Terminal Vertex Gadgets.** For every terminal point $p \in P$ in the RST instance, we place a terminal gadget containing a single blue warrior at its center. As shown in Figure 4.6a, the inclusion constraint forces the wall configuration to enter this gadget and enclose the center, effectively "pinning" the loop to this location.
- **Empty Vertex Gadgets.** For non-terminal grid points, we use an empty vertex gadget to represent them. As shown in Figure 4.6b, an empty vertex gadget allows the wall to pass through (also can work as a steiner point) or bypass the region entirely, as it contains no inclusion units.



(a) Terminal Vertex Gadget (b) Empty Vertex Gadget

Figure 4.6: Vertex Gadgets for RST reduction

4.2.2 Hardness Proof

To establish NP-completeness, we first present a crucial lemma regarding the topological structure of valid wall configurations in the constructed puzzle.

Lemma 2

Any valid wall configuration C in the constructed puzzle must form a simply connected interior region and the set of cells enclosed by C constitutes a **tree polyomino**.

Proof. The proof relies on the dense placement of red warriors at the four corners of every 3×3 gadget (both Terminal and Empty types). Notice that any 2×2 block of cells on the grid must geometrically overlap with at least one corner coordinate of a gadget. Since all gadget corners are occupied by red warriors (which must be in $Ext(C)$), forming a 2×2 block of interior cells would inevitably enclose a red warrior, violating the separation constraint. Consequently, the interior region cannot contain any 2×2 blocks. Combined with the simply-connected property, the solution is strictly mandated to be a tree polyomino. □

Then we can establish NP-hardness by constructing a reduction from Problem 2 using the gadgets described above. Figure 4.7 and Figure 4.8 illustrates a complete instance of this reduction and its solution. The peripheral boundary of the constructed grid is padded with red warriors

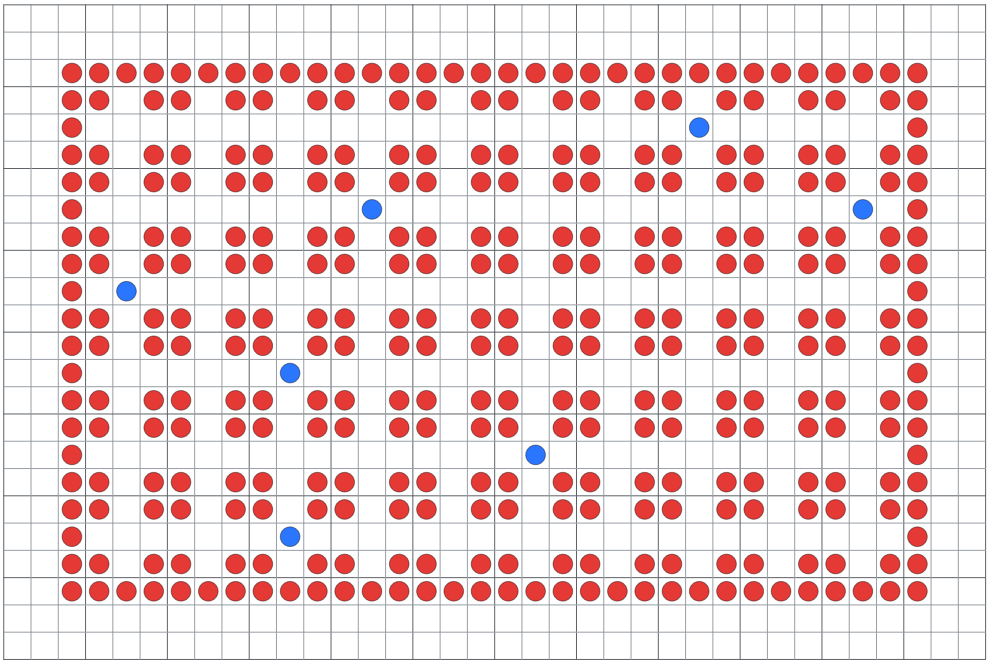


Figure 4.7: W&W from RST Instance

to enforce the wall configuration to remain within the intended gadget area and to establish a consistent exterior region.

Proof. To prove NP-completeness, we demonstrate two properties.

1. Membership in NP As discussed in Section 4.1, a candidate solution of W&W puzzle can be verified in polynomial time. Checking the perimeter length is a simple counting operation $O(|C|)$, and verifying all constraints remains $O(n)$.

Thus, 1-loop W&W puzzle with perimeter bound is in NP.

2. NP-Hardness The relationship between the RST length l and the wall perimeter l' can be established through the topological properties of the enclosed region.

1. **Area-Perimeter Relation:** Let n be the number of unit cells enclosed by the wall configuration. Since the perimeter of any polyomino is determined by the formula $\mathcal{P} = 4n - m$ (where m denotes the number of shared internal edges). This holds because the total initial edge count is $4n$, and each shared internal edge eliminates exactly two

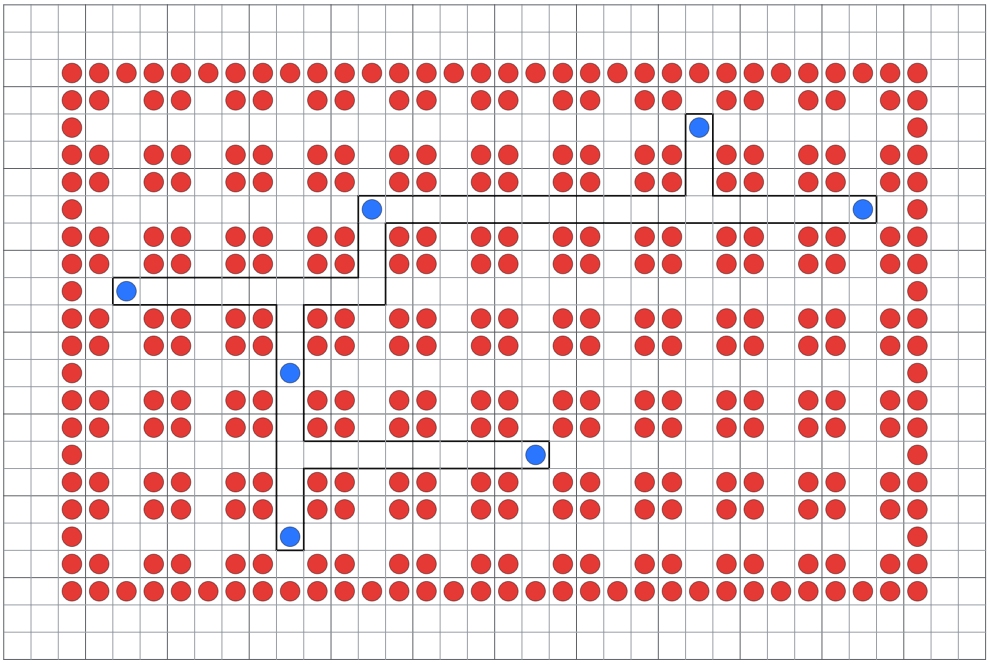


Figure 4.8: The solution of W&W from RST Instance

unit segments from the boundary. Since a tree polyomino of n cells inherently possesses exactly $m = n - 1$ shared edges, its perimeter ℓ' is given by:

$$\ell' = 4n - 2(n - 1) = 2n + 2 \tag{4.1}$$

2. **Mapping Length to Cells:** In our 3×3 grid expansion, a single vertex in the RST is represented by an initial cell ($n = 1$). Each unit of length ℓ in the RST corresponds to an expansion of 3 units in the W&W grid. A single vertex in the original RST problem (where $l = 0$) corresponds to an initial cell of a terminal gadget. Thus, even with zero length, the enclosure must contain 1 initial cell to satisfy the inclusion of the terminal warrior. Therefore, the total number of enclosed cells n for an RST of length ℓ is:

$$n = 3\ell + 1 \tag{4.2}$$

3. **Final Result:** Substituting n into the perimeter formula yields:

$$\ell' = 2(3\ell + 1) + 2 = 6\ell + 4 \tag{4.3}$$

Equation (4.3) demonstrates a bijection between the cost functions.

- **Forward Direction**: If there exists a Steiner Tree of length ℓ, explicitly constructing walls around it yields a valid loop with perimeter $\ell' = 6\ell + 4$.
- **Reverse Direction**: Suppose there exists a valid wall configuration with perimeter $\ell' \leq 6\ell + 4$. By **Lemma 2**, the valid wall configuration strictly encloses a tree polyomino that connects all terminal gadgets. Since the perimeter of a tree polyomino is uniquely determined by its cell count ($\ell' = 2n+2$), we can invert the mapping to find the underlying structure length ℓ'':

$$\ell'' = \frac{\ell' - 4}{6} \tag{4.4}$$

  Given the perimeter constraint $\ell' \leq 6\ell + 4$, it follows that $\ell'' \leq \ell$. This implies the existence of a valid Steiner Tree satisfying the length constraint.

Thus, the W&W instance admits a valid solution under the perimeter constraint if and only if the original RST instance has a Steiner Tree of length at most ℓ. Then the 1-loop W&W puzzle with perimeter bound is NP-hard.

Conclusion Since the problem is in NP and is NP-hard via reduction from RST, the 1-loop W&W puzzle is NP-complete even with strict perimeter bound. □

Chapter 5

Tractability

5.1 Introduction

In the previous chapter, we established that the general 1-loop W&W puzzle is NP-complete (Theorems 1, 2), implying that no polynomial-time algorithm exists for arbitrary grid sizes $H \times W$. However, practical puzzle instances typically operate on a board with a fixed, small height (e.g., $H = 4$) while the width W may vary.

This observation suggests that the hardness of the problem is primarily driven by the grid height. In this chapter, we shift our focus to **Fixed-Parameter Tractability (FPT)**. We propose a **Plug Dynamic Programming (Plug DP)** algorithm that processes the grid vertex-by-vertex. We demonstrate that when the grid height H is bounded by a constant, the problem can be solved in time linear relative to the width W.

5.2 The Plug DP Algorithm

The conceptual foundation of our algorithm is inspired by the techniques developed by Shirayama et al. for Pipe Puzzles [3]. However, significant adaptations were necessary to accommodate the unique constraints of the W&W puzzle, particularly the boundary conditions and the color-based region constraints. Our algorithm scans the grid graph **vertex-by-vertex**. We iterate through the columns j from 0 to W, and within each column, we iterate through the rows i from 0 to H. The pseudo-code for the Plug DP algorithm is as follows:

Algorithm 5.1 Plug DP Algorithm for Walls & Warriors

Input: Grid G of size $H \times W$, Warriors sets B and R.
Output: True if a valid solution exists, False otherwise.

Initialize DP as a map from States to Min-Cost (Boolean).
$State_{start} \leftarrow (P = \{0 \dots 0\}, U = 0, Q = false, C = 0)$
Mark $DP[0][0][State_{start}]$ as True.
for $j = 0$ **to** $W + 1$ **do**
  for $i = 0$ **to** $H + 1$ **do**
    $next\_coord \leftarrow (i + 1 = H)?(0, j + 1) : (i + 1, j)$
    for each $State_{curr}$ in $DP[j][i]$ **do**
      for $w_R \in \{0, 1\}, w_D \in \{0, 1\}$ **do**
        $deg(v) \leftarrow w_L + w_U + w_R + w_D$
        if $deg(v) \in \{0, 2\}$ and Boundary Checks pass **then**
          Update $P_{next}, U_{next}, Q_{next}$ via Union-Find logic
          $C_{next} \leftarrow State_{curr}.C \oplus w_D$
          if Warrior constraints for cell (i, j) satisfied by C_{next} **then**
            $State_{next} \leftarrow$ Normalize$(P_{next}, U_{next}, Q_{next}, C_{next})$
            Mark $DP[next\_coord][State_{next}]$ as True.
          end if
        end if
      end for
    end for
  end for
end for
return True if any $State_{final}$ exists in $DP[W][0]$ where $Q = true$ and P, U are empty;
return False.

5.2.1 The Frontier

The core of the algorithm is the maintenance of a **Frontier**, which is a boundary separating the processed vertices (top-left) from the unprocessed ones (bottom-right). Due to the vertex-by-vertex scan order, the frontier forms a "stepped" line consisting of three distinct segments at step (i, j), as shown in Figure 5.1:

1. **Lower Vertical Frontier** (Rows i to H of column j): This segment separates the processed column $j - 1$ from the current column j. It carries the "Left Walls" (w_L) input for future rows in the current column.
2. **Upper Vertical Frontier** (Rows 0 to $i - 1$ of column $j + 1$): This

segment separates the current column j from the next column $j+1$. It carries the "Right Walls" (w_R) generated by previous steps in the current column.

3. **Horizontal Interaction** (Row i, between columns j and $j+1$): This segment separates the processed vertex $(i-1, j)$ from the current vertex (i, j). It carries the "Up Wall" (w_U) connecting vertically.

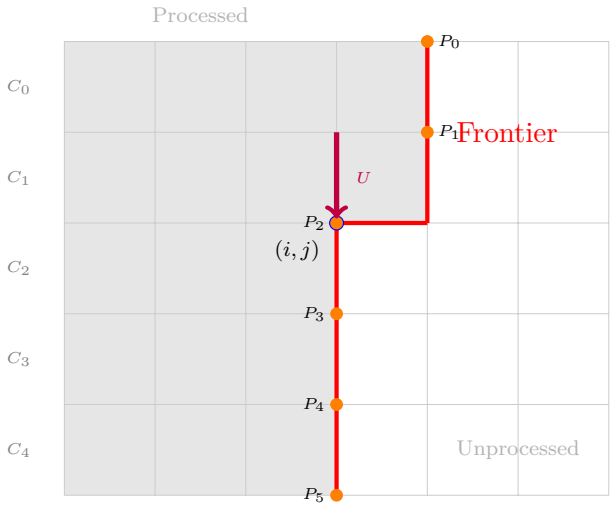


Figure 5.1: Visualization of the Plug DP Frontier at vertex (i, j)

5.2.2 State Definition

A state in our DP must capture all necessary information to ensure global validity (connectivity and loop closure) and local validity (constraints). We define the state signature as a tuple $S = (P, U, C, Q)$:

1. Connectivity: Plugs Array (P)

- **Type**: Integer Array of length $H + 1$.
- **Definition**: P represents the connectivity of the horizontal wall segments crossing the vertical frontiers. $P[k]$ stores the **Component ID** of the wall segment at row k.
- **Normalization**: To minimize the state space, Component IDs are normalized (remapped) to a canonical form (e.g., $1, 2, \ldots$) at each step.

2. Vertical Indicator (U)

- **Type**: Integer.

- **Definition**: U represents the Component ID of the vertical wall segment extending from the vertex directly above $(i-1, j)$ to the current vertex (i, j).
- **Consistency**: This ID shares the same namespace as P. If $U > 0$, it indicates a vertical connection entering the current vertex from above.

3. **Chromatic Status (C)**
   - **Type**: Bitmask of length H.
   - **Definition**: C tracks the geometric "Inside/Outside" status of the regions. The k-th bit corresponds to the region bounded by row k and $k+1$. Value 1 represents Inside the loop (Blue region), and 0 represents Outside the loop (Red region).
   - **Update Logic (XOR)**: The status of a region flips only when a wall boundary is crossed. Specifically, when determining the Down Wall (w_D) at row i:
   $$C_{new}[i] = C_{old}[i] \oplus w_D \tag{5.1}$$
   If $w_D = 1$ (wall placed), the status flips (Inside $\leftrightarrow$ Outside). If $w_D = 0$ (no wall), the status propagates unchanged.

4. **Global Flag (Q)**
   - **Type**: Boolean.
   - **Definition**: Indicates whether a single closed loop has already been fully formed in the processed history.
   - **Constraint**: Once Q becomes true, no active plugs ($P[k] > 0$ or $U > 0$) are allowed in the state, ensuring only one loop exists.

5.2.3 Transitions and Pruning

At each vertex (i, j), the algorithm transitions from a previous state to a new state by deciding the existence of two edges: the **Right Wall (w_R)** and the **Down Wall (w_D)**.

Input:

- **Left Wall (w_L)**: Derived from $P[i]$ of the previous state.
- **Up Wall (w_U)**: Derived from U of the previous state.

Constraints & Conditions:

1. **Degree Constraint**: The number of walls incident to vertex (i, j) must be valid.
$$\deg(v) = w_L + w_U + w_R + w_D \in \{0, 2\} \tag{5.2}$$

(Note: Degree 2 allows for lines and corners. Degree 0 represents an empty vertex).

2. **Connectivity Merge (Union-Find)**: If multiple wall segments meet at (i, j), their component IDs are merged.
   - If merging two different components: Update all occurrences of one ID to the other in P.
   - If merging the same component ID: A loop is closed. We set $Q = \text{true}$. If active plugs remain elsewhere, this branch is invalid (invalid multiple loops).
3. **Warrior Consistency (Color Check)**: For the cell adjacent to the current decision, check the Warrior constraints against the Chromatic Status C.
   - If a blue warrior is at cell (i, j) but $C[i] == 0$ (Outside), prune.
   - If a red warrior is at cell (i, j) but $C[i] == 1$ (Inside), prune.
4. **Cost Optimization**: To handle the optimization variant (finding the minimum wall length), we store the minimum accumulated cost (perimeter) for each unique state signature.

$$DP[S_{new}] = \min(DP[S_{new}], DP[S_{old}] + w_R + w_D) \tag{5.3}$$

5.3 Complexity Analysis

We formally analyze the state space size to establish the FPT property.

- **Connectivity States**: Since the wall configuration is non-self-intersecting, the connections between plugs form a nested, non-crossing structure that is mathematically isomorphic to the balanced parenthesis sequences problem. Consequently, the number of such topological configurations is precisely characterized by the Catalan number $C_{H+1} \approx 4^H$.
- **Chromatic States**: The color mask C has length H, contributing 2^H combinations.
- **Total State Space**: Thus, the maximum number of effective states per frontier is:

$$|S| \approx O(4^H \cdot 2^H) = O(8^H) \tag{5.4}$$

Time Complexity: The algorithm processes $N = W \times H$ vertices. At each vertex, we iterate through all valid states.

$$T(W, H) = O(W \cdot H \cdot 8^H) \tag{5.5}$$

Thus, this complexity is **linear** with respect to the grid width W and exponential only with respect to the fixed parameter H.

Based on the algorithm and analysis above, we present the following theorem:

Theorem 3

The 1-loop Walls & Warriors puzzle is Fixed-Parameter Tractable (FPT) with respect to fixed board height H. Specifically, an instance on a $W \times H$ grid can be solved in $O(W \cdot H \cdot 8^H)$ time.

5.4 Experimental Results

To validate the theoretical bounds, we implemented the Plug DP algorithm in C++ and evaluated its performance on "worst-case" instances (empty grids where the state space is maximized without pruning).

All experiments were conducted on a machine equipped with an Apple M4 chip (10-core CPU) and 16 GB of unified memory, running macOS Sequoia 15.7.3. The algorithms were implemented in C++17 and compiled using Apple Clang 17.0 with the `-O3` optimization flag to ensure maximum performance.

5.4.1 Impact of Grid Height

We fixed the grid width to $W = 10$ and varied the height H from 4 to 8. Table 5.1 summarizes the maximum number of effective states stored in the DP table and the total execution time.

Height (H)	Effective States (Max)	Time (seconds)
4	44	0.0011
5	1,864	0.0245
6	7,640	0.0702
7	29,874	0.2346
8	115,106	0.9807

Table 5.1: Runtime on empty grids with varying height H ($W = 10$).

The results clearly demonstrate the exponential growth of the state space with respect to H. As H increases from 4 to 8, the number of effective states grows by orders of magnitude. However, even at $H = 8$, the problem remains

solvable within 1 second, confirming that the algorithm is highly efficient for the small board heights typical of the puzzle.

5.4.2 Impact of Grid Width

To verify the linear time complexity with respect to the width, we fixed the height to $H = 6$ and varied the width W from 10 to 200.

Width (W)	**Time (seconds)**
10	0.0549
20	0.2034
50	0.6400
100	1.3764
200	2.8995

Table 5.2: Runtime on grids with fixed height $H = 6$ and varying width W.

As shown in Table 5.2, the runtime scales linearly with W. For example, doubling the width from 100 to 200 roughly doubles the execution time (1.37s to 2.90s). This empirical evidence supports the theoretical derivation that the problem can be solved in time linear relative to the width W.

Chapter 6

Conclusion

In this thesis, we have conducted a comprehensive theoretical investigation into the computational complexity of the W&W puzzle. Our research successfully bridges the gap between the physical mechanics of this logic puzzle and the abstract domain of graph theory, establishing both its intractability in the general case and its tractability under structural constraints.

We proved that W&W puzzle is **NP-complete** in the general case by polynomial-time reductions from two NP-complete problems, demonstrating that the puzzle is computationally difficult due to its topological, optimization constraints. However, we identified that the problem becomes tractable when the grid height is bounded: we designed a Fixed-Parameter Tractable (FPT) algorithm using dynamic programming that solves instances with bounded height in linear time relative to width.

In summary, our findings classify W&W puzzle alongside other classic grid-based puzzles such as *Slitherlink* and *Yin-Yang*, satisfying the "good puzzle" property: it is computationally difficult in the general case, ensuring a rich challenge space, yet solvable for bounded instances, allowing for human playability.

Acknowledgment

Pursuing a master's degree in Japan has been a long-held dream of mine. However, when the COVID-19 pandemic disrupted the world between 2020 and 2023, forcing life's gears to a halt, holding onto this ambition required unwavering determination—especially as I entered my early thirties. During this period of isolation, I drew strength from the story of Isaac Newton, who formulated the foundations of his revolutionary scientific ideas while isolated during the Great Plague. While I do not claim to such greatness, I seized that time as a valuable opportunity for self-cultivation. My proficiency in English and Japanese grew significantly, and my foundation in mathematics and algorithms advanced substantially.

These sustained efforts culminated in early 2024 when I was admitted to JAIST. Through a fortunate turn of events, I joined Professor Uehara's laboratory. I would like to express my deepest gratitude to Professor Uehara and Professor Kamata for their exceptional kindness and steadfast support. From the initial uncertainty of selecting a research topic to the eventual progress made through deep academic inquiry, I have gained immense intellectual wealth under their guidance. The study of the Walls & Warriors puzzle has been particularly fascinating, and I am sincerely grateful to Professor Uehara for introducing me to this topic, which has made these two years both intellectually stimulating and personally fulfilling.

To my parents, thank you for your unwavering support of my dreams through all the years leading up to this journey. I also acknowledge my own perseverance in staying the course and overcoming every challenge. Finally, I wish all my fellow students, friends, and esteemed professors the very best in their future endeavors.

Bibliography

[1] R. Uehara, "Computational complexity of puzzles and related topics," *Interdiscip. Inf. Sci.*, vol. 29, no. 2, pp. 119–140, 2023.

[2] A. Itai, C. Papadimitriou, and J. Szwarcfiter, "Hamilton paths in grid graphs," *SIAM J. Comput.*, vol. 11, no. 4, pp. 676–686, 1982.

[3] T. Shirayama, T. Shigemura, Y. Otachi, S. Miyazaki, and R. Uehara, "On computational complexity of pipe puzzles," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E102-A, no. 9, pp. 1134–1141, 2019.

[4] R. A. Hearn and E. D. Demaine, *Games, Puzzles, and Computation.* A K Peters, Ltd., 2009.

[5] H. Wang, "Proving theorems by pattern recognition—II," *Bell Syst. Tech. J.*, vol. 40, no. 1, pp. 1–41, 1961.

[6] R. Berger, "The undecidability of the domino problem," *Mem. Amer. Math. Soc.*, no. 66, pp. 1–72, 1966.

[7] SmartGames. (2026) Walls & warriors. [Accessed: Jan. 5, 2026]. [Online]. Available: https://www.smartgamesusa.com/one-player-games/walls-warriors

[8] T. Yato and T. Nagata, "Complexity and completeness of finding another solution and its application to puzzles," *IEICE Trans. Inf. Syst.*, vol. E86-D, no. 5, pp. 1052–1060, 2003.

[9] E. D. Demaine, J. Lynch, M. Rudoy, and Y. Uno, "Yin-Yang puzzles are NP-complete," *CoRR*, vol. abs/2106.15585, 2021.

[10] E. D. Demaine and M. Rudoy, "Tree-residue vertex-breaking: A new tool for proving hardness," 2018.

[11] M. R. Garey and D. S. Johnson, "The rectilinear Steiner tree problem is NP-complete," *SIAM J. Appl. Math.*, vol. 32, no. 4, pp. 826–834, 1977.

[12] A. Papakostas and I. G. Tollis, "Algorithms for area-efficient orthogonal drawings," *Comput. Geom.*, vol. 9, no. 1-2, pp. 83–110, 1998.